Тема 1: Технологія розробки комп'ютерних програм

План

1. Архітектура цифрового комп'ютера
2. Подання даних
3. Етапи розв'язку задач на комп'ютері.
4. Поняття алгоритму.
5. Властивості алгоритму.
6. Форми представлення алгоритму.
7. Базові структури алгоритмів.

Література

- Интерактивный учебник языка Python. http://pythontutor.ru/ .
- **Златопольский Д. М. Основы программирования на языке Python.**
- **Федоров Д. Ю. Программирование на языке высокого уровня Python**
- **Dr. Charles R. Severance Python for Everybody Exploring Data Using Python 3**
- **Яковенко А. В. Основи програмування. Python. Частина 1; КПІ ім. Ігоря Сікорського.**
- Програмування числових методів мовою Python : навч. посібник.
- Бардачов Ю. М., Соколова Н. А., Ходаков В. Є. Дискретна математика.
- Хахаев И. А. Практикум по алгоритмизации и программированию на Python.

- Робота на лекції 2 бали
- Лабораторні заняття 7 50 балів
- Домашні завдання 2 8 балів
- Модульні контрольні роботи 2*20 40 балів

  ----

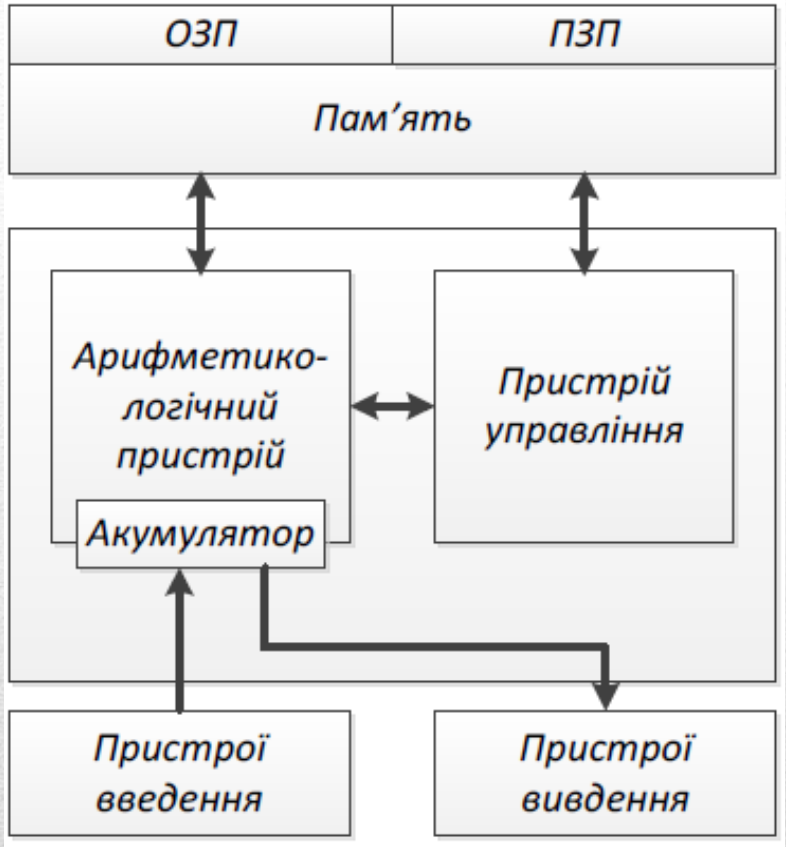
  100 балів
- **Бонус** – за умови відсутності пропусків лекцій – 2 бали

Критерії оцінювання знань студентів

1. Архітектура цифрового комп'ютера

- ***ЕОМ – електронна обчислювальна машина***, що здатна виконувати обчислювальні задачі будь-якої складності за наявності відповідної програми.
- Усе забезпечення комп'ютера ділиться на дві частини: ***апаратну*** та ***програмну***.
- До ***апаратного*** забезпечення відносяться пристрої і прилади, що утворюють апаратну конфігурацію.
- Сучасні комп'ютери й обчислювальні комплекси мають ***блоково-модульну конструкцію***. Основний блок персонального комп'ютера (ПК) – системний блок. Він містить блок електроживлення, кріпильні елементи для материнської (системної) плати, електронних плат і дисководів.

- ***Архітектурою комп'ютера*** називають склад і взаємозв'язок основних пристроїв комп'ютера.
- Для того, щоб комп'ютер був ефективним і універсальним інструментом він має включати такі компоненти:
- – арифметико-логічний пристрій;
- – пристрій управління;
- – запам'ятовуючий пристрій чи пам'ять;
- – пристрої уведення-виведення інформації.



Архітектура комп’ютера (фон Неймана)

- **Арифметико-логічний пристрій** виконує арифметичні та логічні перетворення даних, що надходять до нього.
- **Пристрій управління** автоматично керує процесом оброблення інформації, посилаючи всім іншим пристроям сигнали про виконання тих чи інших дій.
- Сукупність арифметико-логічного пристрою та пристрою управління називають **процесором** або CPU (Central Processing Unit).
- **Акумулятор** (або регістр) також є частиною процесора і слугує для збереження проміжних результатів обчислень, щоб не звертатися кожного разу до основної пам’яті.

Пам’ять

- **Пам’ять** зберігає програми та дані, що передані з інших пристроїв (зокрема, пристроїв уведення), і видає інформацію іншим пристроям комп’ютера, включаючи пристрої виведення.
- Там же зберігаються всі проміжні дані, необхідні для виконання програми. Пам’ять складається з великої кількості комірок, кожна з яких має адресу і містить 1 байт (тобто 8 біт інформації).
- Це дозволяє в будь-який момент часу звернутися до будь-якої комірки і зчитати або записати туди дані. Але дані зберігаються лише поки наявне живлення, а при знеструмленні комп’ютера всі данні стираються. Саме тому ми знаємо її як **оперативну пам’ять** або RAM (random access memory).

- Для того, щоб дані нікуди не зникали після завершення роботи комп’ютера, використовуються додаткові запам’ятовуючі пристрої з нижчою швидкістю доступу до інформації.
- Найбільш розповсюдженим варіантом є **жорсткий диск** – на якому зберігаються всі аудіо-, відео- та ін. файли та записана операційна система і всі інші програми. Під час запуску програми вона копіюється у оперативну пам’ять і далі виконується звідти. Всі дані, з якими працює програма, також завантажуються з жорсткого диска та назад за вимогою. Саме тому, при вимкненні комп’ютера слід зберегти всі відкриті документи – в цей час всі зміни існують лише в оперативній пам’яті і зникають з неї разом із струмом.
- Жорсткий диск, як і всі інші пристрої, що підключаються до комп’ютера, відноситься до **пристроїв вводу-виводу**.
- Кожен з них має власний набір команд для взаємодії з комп’ютером і потребує для роботи спеціальну програму, що називається **драйвером**.
- Якщо в операційній системі встановлений відповідний драйвер, вона здатна "спілкуватися" з підключеним пристроєм, що дозволяє його використовувати.

Жорсткий диск

- Зазвичай вхідні і вихідні дані подаються у формі, зручній для людини. Числа люди звикли зображати в **десятковій** системі числення.
- Для комп'ютера зручніше **двійкова** система. Це пояснюється тим, що технічно набагато простіше реалізувати пристрої (наприклад, запам'ятовуючий елемент) з двома, а не з десятьма стійкими станами (є електричний струм – немає струму, намагнічений – не намагнічений і т.п.). Можна вважати, що один з двох станів означає одиницю, інший – нуль.
- Будь-які дані (числа, символи, графічні та звукові образи) в комп'ютері представляються у вигляді послідовностей з нулів і одиниць. Ці послідовності можна вважати словами в алфавіті {0,1}, так що оброблення даних всередині комп'ютера можна сприймати як перетворення слів з нулів і одиниць за правилами, зафіксованим в мікросхемах процесора.
- Зазвичай код символу зберігається в одному байті. Код символу розглядається як число без знаку і, отже, може набувати значень від 0 до 255. Такі кодування називають однобайтовими; вони дозволяють використовувати до 256 різних символів.

2. Подання даних

1. Постановка задачі
2. Побудова математичної моделі
3. Розробка алгоритму
4. Складання програми
5. Компіляція програми
6. Компонування програми
7. Налагодження програми
8. Експлуатація програми

3. Етапи розв'язку задач на комп'ютері.

Аль Хорезмі (783-850)

- Походження слова «алгоритм» - від імені арабського вченого **Аль Хорезмі,** який приблизно у 825 році дав опис винайденій в Індії позиційній десятковій системі числення. У перекладі книжка містила ім‘я вченого, від якого пішло слово «алгоритм», а від оригінальної назви пішло слово «алгебра». Саме він у своїх трактатах описав правила (алгоритми) додавання, віднімання, множення та ділення багатозначних чисел, якими ми користуємося сьогодні.
- В 1936 англійський математик **Алан Тьюрінг** запропонував математичну модель обчислювальної машини, відому як машина Тьюрінга. Крім того, довів, що ця машина здатна виконати будь-які обчислення, і ввів поняття алгоритму – як послідовності дій, необхідних машині для досягнення результату. Сама ж комп’ютерна програма являє собою записаний набір команд, які задають алгоритм у формі, зрозумілій для машини.
- Для того, щоб навчити комп’ютер щось робити, потрібно попередньо скласти **алгоритм.**

4. Поняття алгоритму

Визначення

Алгоритм – скінчений набір правил, який визначає послідовність операцій для розв’язку конкретної множини задач та володіє наступними важливими рисами: скінченістю, визначеністю, вводом, виводом, ефективністю.

Алгоритм – система обчислень, що виконується за чітко визначеними правилами, яка після деякої кількості кроків приводить до розв’язку поставленої задачі.

Алгоритм – чітко детермінована послідовність дій, яка описує процес перетворення об’єкту із початкового стану в кінцевий, записана за допомогою зрозумілих виконавцю команд.

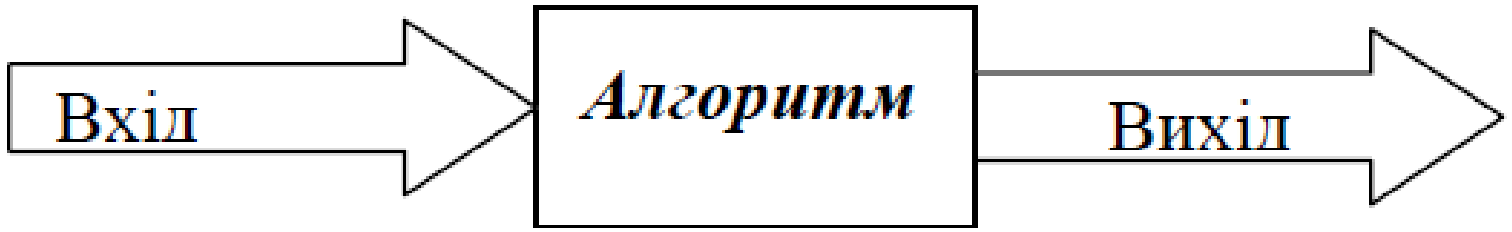
Визначення

- **Алгоритм** - послідовність дій, направлених на отримання кінцевого результату за скінчену кількість кроків.
- **Алгоритм** - послідовність дій, яка або приводить до розв’язку задачі, або пояснює, чому такий розв’язок отримати не можливо.
- **Алгоритм** - точна, однозначна, скінчена послідовність дій, яку необхідно виконати для досягнення конкретної мети за скінчену кількість кроків.
- **Алгоритм** — це скінчена послідовність команд, які потрібно виконати над вхідними даними для отримання результату.

Алгоритм описує процес перетворення вихідних даних у результат, оскільки для розв’язку будь-якої задачі необхідно:

1. Ввести вихідні дані.
2. Перетворити вихідні дані у результат (вихідні дані).
3. Вивести результат.

Алгоритм – це деяке правило перетворення інформації, застосування якого до заданої (вихідної) інформації приводить до результату – нової інформації.



Алгоритм як правило перетворення інформації

Запис алгоритму розпадається на окремі вказівки виконавцю виконати деяку закінчену дію.

Кожна така вказівка називається **командою**.

Команди алгоритму виконуються послідовно одна за одною.

Після кожного кроку виконання алгоритму точно відомо, яка команда повинна виконуватися наступною.

Сукупність команд, які можуть бути виконані виконавцем, називається **системою команд виконавця**.

Вихідна інформація *A, B, C, D; {скласти A/B і C/D}*
Результат *E,F;* *{E/F = A/B + C/D}*

1. Обчислити Y = B*D;	{Перейти до наступної команди}
2. Обчислити X_1 = A*D;	{Перейти до наступної команди}
3. Обчислити X_2 = B*C;	{Перейти до наступної команди}
4. Обчислити X = X_1+X_2;	{Перейти до наступної команди}
5. Обчислити Z = НСД (X,Y);	{Перейти до наступної команди}
6. Обчислити E = X div Z;	{Перейти до наступної команди}
7. Обчислити F = Y div Z.	{Закінчити роботу}.

Приклад алгоритму додавання дробів

- *Масовість.* Алгоритм повинен бути застосованим до будь – яких елементів з множини вихідних даних.
- *Визначеність.* Операції, які використовуються в алгоритмі, не повинні мати двоякого тлумачення; не повинно виникати питання: що саме і як треба робити? Порядок виконання операцій повинен бути чітко визначеним.
- *Дискретність.* Процес розв'язування алгоритму повинен складатися з окремих завершених операцій, які виконуються послідовно і за скінчений час.
- *Результативність.* Виконання послідовності операцій алгоритму повинно приводити до цілком конкретного результату.
- *Формальність.* Будь – який виконавець, здатний сприймати і виконувати вказівки алгоритму (навіть не розуміючи їх змісту), діючи за алгоритмом, може виконати постановлене завдання.
- *Елементарність.* Кожна команда з набору команд Виконавця містить вказівку виконати деяку елементарну (не деталізовану більш детально) дію, яку розуміє, однозначно і точно виконує Виконавець.

5. Властивості алгоритму.

- *словесна форма* – це запис алгоритму у вигляді послідовності занумерованих словесних команд;
- *таблична форма* подання алгоритму – це запис алгоритму у вигляді розрахункової таблиці;
- *блок-схема* – це запис алгоритму у графічній формі з використанням спеціальних геометричних фігур (блоків), які містять опис операцій, і направлених ліній, які показують напрями передавання управління від одного блоку до іншого.
- *алгоритмічна мова* – це штучна мова, призначена для подання алгоритмів;
- *псевдокод* – опис структури алгоритму природною, частково-формалізованою мовою, що дозволяє виявити основні етапи рішення задачі перед точним його записом на мові програмування;
- *мова програмування* – це алгоритмічна мова, конструкції якої однозначно перетворюються на команди комп'ютеру. Програма – це алгоритм, записаний на мові програмування.

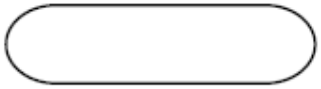
6. Форми представлення алгоритму.

Задача: знайти модуль величини x (тобто значення |x|) і присвоїти це значення змінній y. Під час побудови алгоритму скористаємося визначенням модуля: | x| = x при x ≥ O і |x| = -x при x < 0.

- Алгоритм можна записати наступним чином
- 1 . Початок.
- 2. Ввести числове значення величини x.
- 3. Якщо x ≥ 0, то y присвоїти значення x, інакше y присвоїти значення -x.
- 4. Вивести значення y.
- 5. Кінець.

Приклад опису алгоритму у словесній формі.

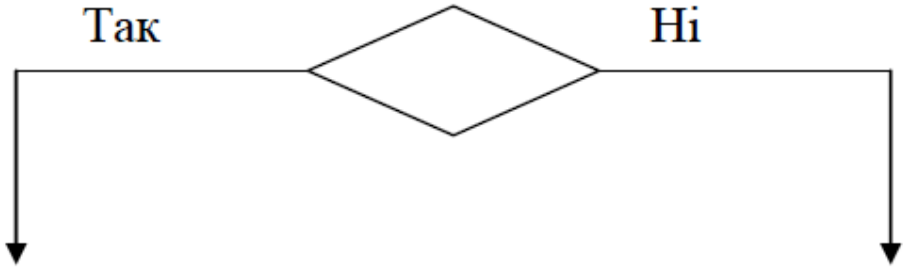
1. Блок початку або кінця алгоритму:



2. Блоки введення- виведення:

3. Командний (операторний) блок:

4. Умовний блок:

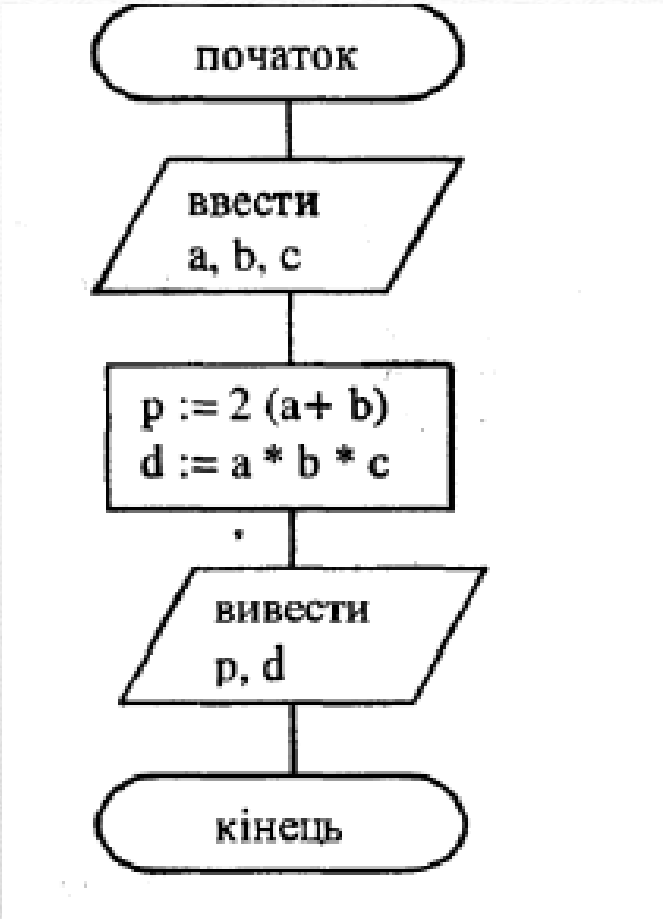


5. Початок циклічного обчислення. Умовний блок

6. Обчислення за підпрограмою:

7. Розрив лінії обчислювального потоку:

Блок схема складається з блоків декількох видів



Приклад запису блок-схеми алгоритму розв'язку задачі

Усі команди записують у прямокутних блоках, один на одному.

Порядок розміщення блоків визначає порядок виконання команд.

ввести a,b,c
p:=2*(a+b)
d:= a* b* c
вивести p,d

Приклад опису алгоритму за допомогою структурної схеми

Приклад опису алгоритму за допомогою псевдокоду

Алгоритм заповнення залікової відомості групи з 20 студентів (і - номер студента).

псевдокод

початок циклу

для

і від 1 до 20 з кроком 1

повторювати:

1.1. ввести прізвище студента

1.2. поставити оцінку.

кінець циклу (КЦ)

виведення відомості

Існують такі правила графічного запису алгоритмів:

- блоки алгоритмів з’єднуються лініями потоків інформації;
- лінії потоків не повинні перетинатися;
- будь-який алгоритм може мати лише один блок початку і один блок кінця.

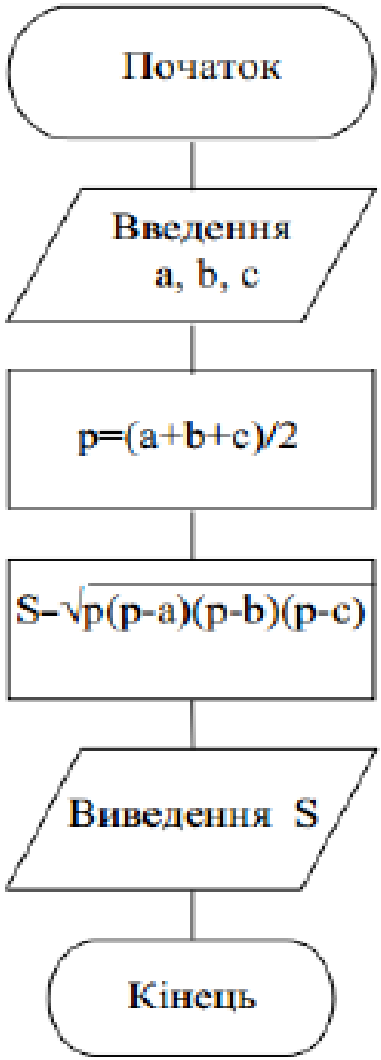
Можна виділити всього чотири базові структури:

- лінійні;
- розгалужені;
- циклічні;
- змішані.

7. Базові структури (алгоритмічні конструкції) алгоритмів.

ЛІНІЙНІ АЛГОРИТМИ

Лінійним називається алгоритм (фрагмент алгоритму), в якому окремі команди виконуються послідовно друг за другом, не залежно від значень вхідних даних і проміжних результатів



. Блок-схема алгоритму обчислення площини трикутника

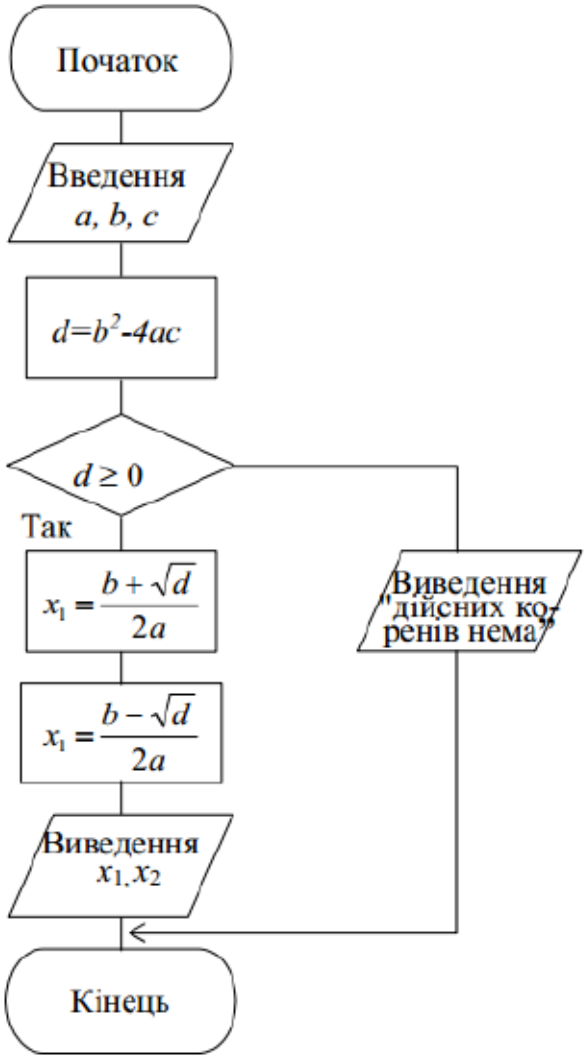
РОЗГАЛУЖЕНИЙ АЛГОРИТМ

Часто хід обчислювального процесу залежить від вихідних даних або проміжних результатів; алгоритм реалізується в одному з декількох, заздалегідь передбачених (можливих) напрямків.

Такі напрямки часто називаються гілками. Кожна гілка може бути будь-якого ступеня складності, а може взагалі не містити команд, тобто бути виродженою.

Вибір тієї або іншої гілки здійснюється в залежності від результату перевірки умови з конкретними даними.

У кожному випадку алгоритм реалізується тільки по одній гілці, а виконання інших виключається.



Блок-схема алгоритму обчислення дійсних коренів квадратичного рівняння

ЦИКЛІЧНИЙ АЛГОРИТМ

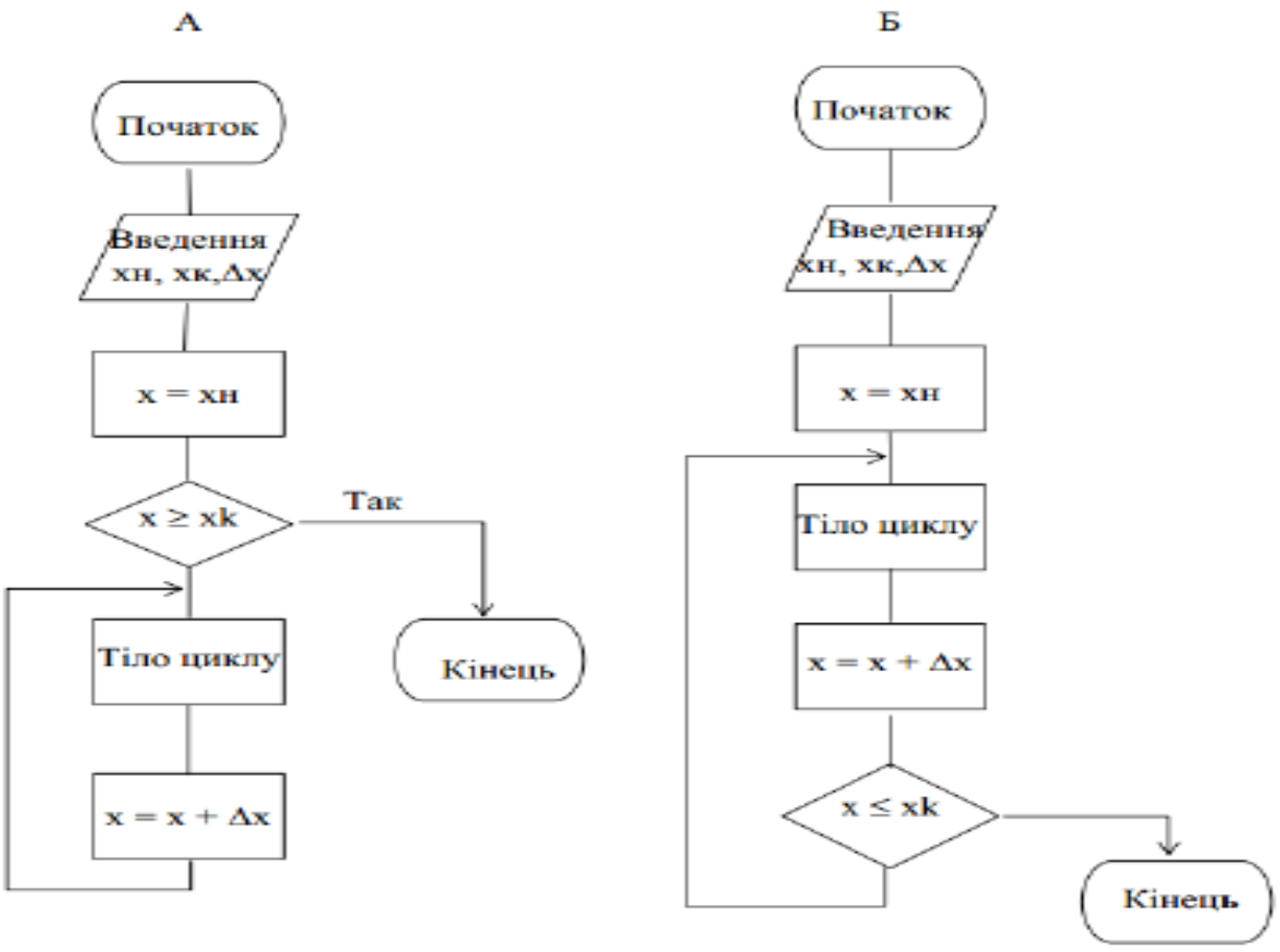
У залежності від способу організації кількості повторень циклу розрізняють три типи циклів:

1) цикл із заданою умовою закінчення роботи (ЦИКЛ - ДО);

2) цикл із заданою умовою продовження роботи (ЦИКЛ - ПОКИ);

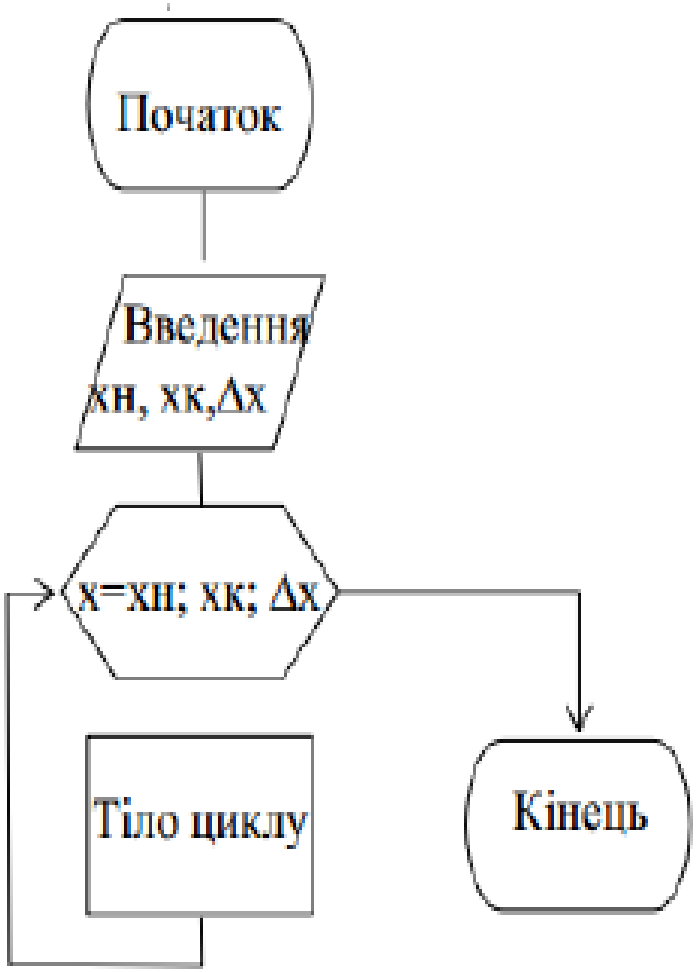
3) цикл із заданою умовою повторень роботи (ЦИКЛ З ПАРАМЕТРОМ).

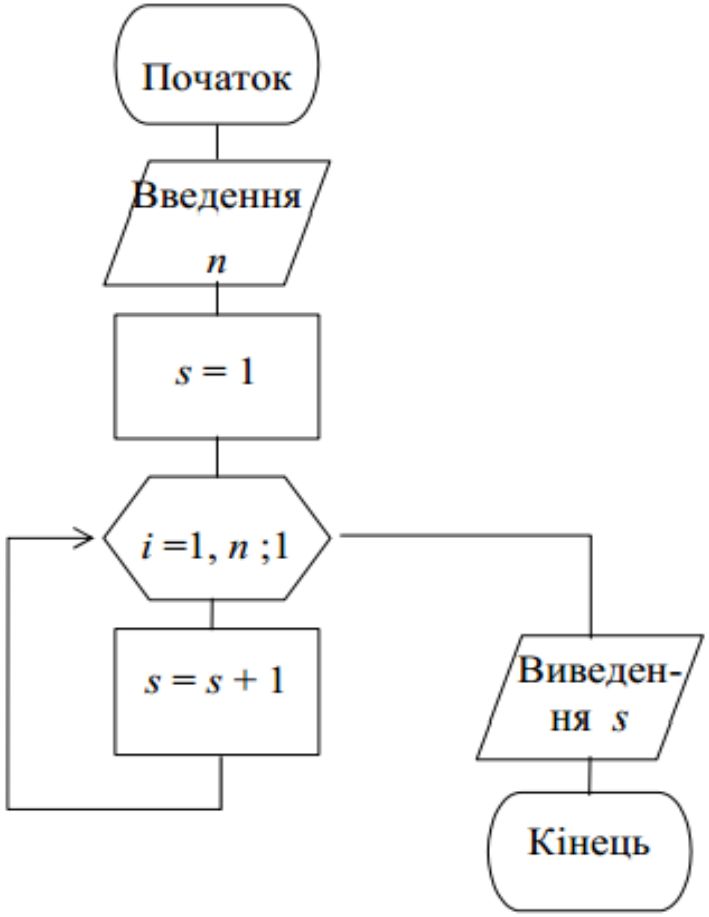
ЦИКЛІЧНИЙ АЛГОРИТМ



Блок-схема організації циклу
А-цикл із заданою умовою закінчення роботи (цикл - до);
Б-цикл із зданою умовою продовження роботи (цикл – поки)

ЦИКЛ З ПАРАМЕТРОМ



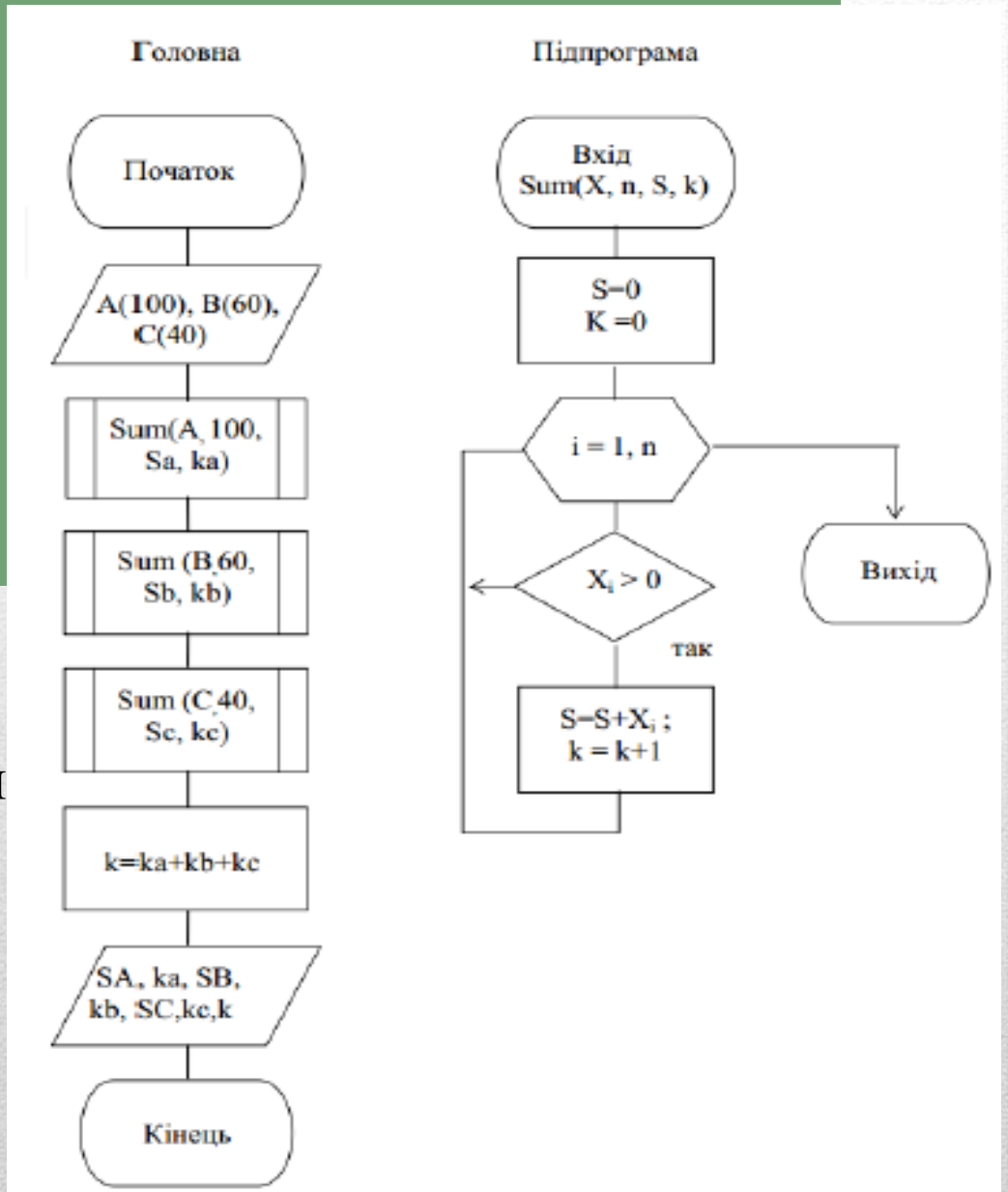


Блок-схема обчислення суми перших n чисел

ПРОЦЕДУРА

Процедура - це спеціальним образом оформлена послідовність команд, до якої можна звернутися: зажадати перервати виконання команд основної програми, виконати всю послідовність команд процедури, а потім продовжити виконання команд основної програми.

В якості процедури можуть використовуватись зовнішні функції (підпрограми-функції) і зовнішні підпрограми.



Приклад алгоритму пошуку суми та кількості додатних елементів масивів A, B і C з використанням підпрограми

Дякую за увагу!!!